


```
CCCCCCCC 000000 MM MM RRRRRRRR 55555555 000000 WW WW DDDDDDDD
CCCCCCCC 000000 MM MM RRRRRRRR 55555555 000000 WW WW DDDDDDDD
CC 00 00 MMMM MMMM RR RR 55 00 00 WW WW DD DD
CC 00 00 MMMM MMMM RR RR 55 00 00 WW WW DD DD
CC 00 00 MM MM MM RR RR 555555 00 0000 WW WW DD DD
CC 00 00 MM MM MM RR RR 555555 00 0000 WW WW DD DD
CC 00 00 MM MM MM RRRRRRRR 55 00 00 WW WW DD DD
CC 00 00 MM MM MM RRRRRRRR 55 00 00 WW WW DD DD
CC 00 00 MM MM RR RR 55 0000 00 WW WW DD DD
CC 00 00 MM MM RR RR 55 0000 00 WW WW DD DD
CC 00 00 MM MM RR RR 55 00 00 WWW WW DD DD
CC 00 00 MM MM RR RR 55 00 00 WWW WW DD DD
CCCCCCCC 000000 MM MM RR RR 555555 000000 WW WW DDDDDDDD
CCCCCCCC 000000 MM MM RR RR 555555 000000 WW WW DDDDDDDD
.....
.....
.....
.....
```

```
LL IIIII SS SSSSSSS
LL IIIII SS SSSSSSS
LL II SS
LL II SS
LL II SS
LL II SSSSSS
LL II SSSSSS
LL II SS
LL II SS
LL II SS
LL II SS
LLLLLLLLL IIIII SSSSSSS
LLLLLLLLL IIIII SSSSSSS
```


(2) 50
(3) 71
(4) 99

HISTORY : Detailed Current Edit History
DECLARATIONS
COMSSR50WD_R6 - CONVERT 3 ASCII CHARS INTO ONE WORD RADIX-50 VALUE


```
0000 1 .TITLE COM$R50WD ; FORTRAN COMPATIBILITY - ASCII TO RADIX-50 CONVERSI
0000 2 .IDENT /1-004/ ; File: COMR50WD.MAR
0000 3
0000 4
0000 5 *****
0000 6 *
0000 7 * COPYRIGHT (c) 1978, 1980, 1982, 1984 BY *
0000 8 * DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS. *
0000 9 * ALL RIGHTS RESERVED. *
0000 10 *
0000 11 * THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED *
0000 12 * ONLY IN ACCORDANCE WITH THE TERMS OF SUCH LICENSE AND WITH THE *
0000 13 * INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR ANY OTHER *
0000 14 * COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY *
0000 15 * OTHER PERSON. NO TITLE TO AND OWNERSHIP OF THE SOFTWARE IS HEREBY *
0000 16 * TRANSFERRED. *
0000 17 *
0000 18 * THE INFORMATION IN THIS SOFTWARE IS SUBJECT TO CHANGE WITHOUT NOTICE *
0000 19 * AND SHOULD NOT BE CONSTRUED AS A COMMITMENT BY DIGITAL EQUIPMENT *
0000 20 * CORPORATION. *
0000 21 *
0000 22 * DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE OR RELIABILITY OF ITS *
0000 23 * SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL. *
0000 24 *
0000 25 *
0000 26 *****
0000 27
0000 28
0000 29 FACILITY: FORTRAN COMPATABILITY LIBRARY
0000 30 ++
0000 31 ABSTRACT:
0000 32
0000 33 COM$R50WD_R6 performs conversion of 3 ASCII characters to 1 word.
0000 34 It is used by FORTRAN compatibility routines RAD50 and IRAD50.
0000 35
0000 36 --
0000 37
0000 38 VERSION: 0
0000 39
0000 40 HISTORY:
0000 41
0000 42 AUTHOR:
0000 43 Peter Yuo, 12-Sep-77: Version 0
0000 44
0000 45 MODIFIED BY:
0000 46
0000 47
0000 48
```



```
0000 50      .SBTTL HISTORY      ; Detailed Current Edit History
0000 51
0000 52
0000 53 : Edit History for Version 01 of ASCR50
0000 54 :
0000 55 : 0-03 Clear RADIX_VALUE at initialization in R50WD_R6
0000 56 : 00-06 - Define formal for RAD50 so no access vio. TNH 5-Jan-78
0000 57 : 00-07 - Make PSECT be F4PCOMPAT$CODE. TNH 5-Jan-78
0000 58 : 0-8 - Bug fix for RAD50. JMT 5-Jan-78
0000 59 : 0-9 - Another bug fix for RAD50. JMT 9-Jan-77
0000 60 : 1-1 - Break module COM$ASCR50 into 3 modules:
0000 61 :          COM$RAD50 - routine RAD50
0000 62 :          COM$IRAD50 - routine IRAD50
0000 63 :          COM$SR50WD - common ASCII to RAD50 conversion routine
0000 64 : 1-002 - Update copyright notice. JBS 16-NOV-78
0000 65 : 1-003 - Add "" to PSECT directive. JBS 21-DEC-78
0000 66 : 1-004 - Blanks were not being counted as characters converted.
0000 67 :          Also, routine did not stop at first non-rad50 char as
0000 68 :          specified. Recode to conform with PDP-11.
0000 69 :          SPR 11-26803 SBL 31-Oct-1979
```



```

0000 71      .SBTTL  DECLARATIONS
0000 72
0000 73 :
0000 74 : INCLUDE FILES:
0000 75 :
0000 76 :
0000 77 :
0000 78 : EXTERNAL SYMBOLS:
0000 79 :
0000 80      .DSABL  GBL
0000 81
0000 82 :
0000 83 : MACROS:
0000 84 :
0000 85 :
0000 86 :
0000 87 : PSECT DECLARATIONS:
0000 88 :
00000000 89      .PSECT  _F4PCOMPAT$CODE PIC,USR,CON,REL,LCL,SHR,EXE,RD,NOWRT
0000 90
0000 91 :
0000 92 : EQUATED SYMBOLS:
0000 93 :
0000 94 :
0000 95 :
0000 96 : OWN STORAGE:
0000 97 :

```



```
0000 99 .SBTTL COM$R50WD_R6 - CONVERT 3 ASCII CHARS INTO ONE WORD RADIX-50 VALUE
0000 100
0000 101
0000 102
0000 103 :++
0000 104 : FUNCTIONAL DESCRIPTION:
0000 105 :
0000 106 : Algorithmic steps:
0000 107 : 1) Initialization
0000 108 : REM_CHAR_IN_WORD = 3 (3 radix50 chars/word)
0000 109 : 2) If (CHARS_REM - 1) < 0 then (CURRENT_CHAR = 0, go to
0000 110 : step 5 to fill up the rest of the word)
0000 111 : otherwise (CURRENT_CHAR = CHAR(NEXT_INPUT_POSITION),
0000 112 : NEXT_INPUT_POSITION = NEXT_INPUT_POSITION + 1).
0000 113 : 3) Get the corresponding radix-50 value
0000 114 : a. If ASCII('A') <= ASCII(CURRENT_CHAR) <= ASCII('Z') then
0000 115 : CURRENT_CHAR = ASCII(CURRENT_CHAR) - 100(octal)
0000 116 : b. If ASCII('0') <= ASCII(CURRENT_CHAR) <= ASCII('9') then
0000 117 : CURRENT_CHAR = ASCII(CURRENT_CHAR) - 22(octal)
0000 118 : c. If ASCII(CURRENT_CHAR) = ASCII(' ') then
0000 119 : CURRENT_CHAR = ASCII(' ') - 40(octal)
0000 120 : d. If ASCII(CURRENT_CHAR) = ASCII('$') then
0000 121 : CURRENT_CHAR = ASCII('$') - 11(octal)
0000 122 : e. If ASCII(CURRENT_CHAR) = ASCII('.') then
0000 123 : CURRENT_CHAR = ASCII('.') - 22(octal)
0000 124 : f. If none of the above then terminate.
0000 125 : ADD COUNT:
0000 126 : 4) ACTUAL_CHAR_COUNT = ACTUAL_CHAR_COUNT + 1
0000 127 : ACCUM:
0000 128 : 5) RADIX_VALUE = RADIX_VALUE * 50(octal) + CURRENT_CHAR
0000 129 : 6) If (CHARS_REM = CHARS_REM - 1) > 0 THEN go back to step 2
0000 130 : 7) return with the result in RADIX-VLAUE
0000 131 :--
0000 132 COM$R50WD_R6::
0000 133 :
0000 134 : Initialization
0000 135 :
0000 136 :
56 03 D0 0000 137 MOVL #3, R6 ; R6 = CHARS_REM_IN_WORD = 3
51 D4 0003 138 CLRL R1 ; clear RADIX_VALUE
0005 139
0005 140 :
0005 141 : Clear CHARRENT_CHAR
0005 142 : If (CHARS_REM = 1) <= 0 then (CURRENT_CHAR = 0, go to ACCUM to fill
0005 143 : up the rest of the word)
0005 144 : else (CURRENT_CHAR = CHAR(NEXT_INPUT_POSITION),
0005 145 : NEXT_INPUT_POSITION = NEXT_INPUT_POSITION + 1)
0005 146 :
53 D4 0005 147 AGAIN1: CLRL R3 ; clear CURRENT_CHAR
55 D7 0007 148 DECL R5 ; CHARS_REM = CHARS_REM - 1
46 19 0009 149 BLSS ACCUM ; branch to fill up the rest of the word
53 82 9A 000B 150 MOVZBL (R2)+, R3 ; CURRENT_CHAR = next input char
000E 151 ; and advance NEXT_INPUT_POSITION
000E 152
000E 153 :
000E 154 : Get the corresponding RADIX-50 value
000E 155 : a. If ASCII(CURRENT_CHAR) <= ASCII(' ') then go to SPACE
```



```
000E 156 : b. If ASCII(CURRENT_CHAR) > ASCII('Z') then terminate scan
000E 157 : c. If ASCII(CURRENT_CHAR) < ASCII('A') then go to CHECK_NUMBER
000E 158 : d. current char is A-Z, so CURRENT_CHAR = CURRENT_CHAR - 100(octal)
000E 159 : go to ACCUM
000E 160 : CHECK_NUMBER:
000E 161 : e. If ASCII(CURRENT_CHAR) < ASCII('0') then go to CHECK_DOLLAR
000E 162 : f. If ASCII(CURRENT_CHAR) > ASCII('9') then terminate scan
000E 163 : g. current char is 0-9, so CURRENT_CHAR = CURRENT_CHAR - 22(octal)
000E 164 : go to ACCUM
000E 165 : CHECK_DOLLAR:
000E 166 : h. If ASCII(CURRENT_CHAR) = ASCII('$') then (CURRENT_CHAR = 33(octal),
000E 167 : go to ACCUM)
000E 168 : i. If ASCII(CURRENT_CHAR) = ASCII('.') then (CURRENT_CHAR = 34(octal),
000E 169 : go to ACCUM)
000E 170 :
000E 171 :
20 53 91 000E 172 CMPB R3, #^A/ / : compare CURRENT_CHAR with space
3A 13 0011 173 BEQL SPACE : a space character?
4F 19 0013 174 BLSS ILLEGAL : not a RAD50 character
5A 8F 53 91 0015 175 CMPB R3, #^A/Z/ : compare CURRENT_CHAR with 'Z'
49 14 0019 176 BGTR ILLEGAL : not a RAD50 character
41 8F 53 91 001B 177 CMPB R3, #^A/A/ : compare CURRENT_CHAR with 'A'
09 19 001F 178 BLSS CHECK_NUMBER : branch to check if CURRENT_CHAR is
0021 179 : a number
53 00000040 8F C2 0021 180 SUBL #^0100, R3 : R3 = correspondin radix-50 value
0028 181 : for A-Z
25 11 0028 182 BRB ADD_COUNT : branch to add acutal count
002A 183 CHECK_NUMBER:
30 53 91 002A 184 CMPB R3, #^A/0/ : compare CURRENT_CHAR with '0'
0A 19 002D 185 BLSS CHECK_DOLLAR : go to check for '$'
39 53 91 002F 186 CMPB R3, #^A/9/ : compare CURRENT_CHAR with '9'
30 14 0032 187 BGTR ILLEGAL : Not a RAD50 character
53 12 C2 0034 188 SUBL #^022, R3 : get corresponding radix-50 value
0037 189 : for 0-9
16 11 0037 190 BRB ADD_COUNT : branch to add actual count
0039 191 CHECK_DOLLAR:
24 53 91 0039 192 CMPB R3, #^A/$/ : compare CURRENT_CHAR with '$'
05 12 003C 193 BNEQ CHECK_PERIOD : branch to check for period
53 1B D0 003E 194 MOVL #^033, R3 : CURRENT_CHAR = corresponding
0041 195 : radix-50 value
0C 11 0041 196 BRB ADD_COUNT : branch to add to ACUTAL_COUNT
0043 197 CHECK_PERIOD:
2E 53 91 0043 198 CMPB R3, #^A/./ : compare CURRENT_CHAR with '.'
1C 12 0046 199 BNEQ ILLEGAL : not a RAD50 character
53 1C D0 0048 200 MOVL #^034, R3 : get corresponding radix-50 value
02 11 004B 201 BRB ADD_COUNT : branch to add ACTUAL_COUNT
004D 202 SPACE:
53 D4 004D 203 CLRL R3 : CURRENT_CHAR = 0
004F 204
004F 205
004F 206 :
004F 207 : Accumulate ACTUAL_COUNT
004F 208 :
004F 209
50 D6 004F 210 ADD_COUNT:
0051 211 INCL R0 : ACTUAL_COUNT = ACTUAL_COUNT + 1
0051 212
```



```
0051 213 :  
0051 214 : Accumulate ACTUAL_VALUE  
0051 215 :  
0051 216 :  
51 51 03 78 0051 217 ACCUM: ASHL #3, R1, R1 ; R1 = 8*RADIX_VALUE  
51 53 51 C0 0055 218 ADDL R1, R3 ; R3 = 8*RADIX_VALUE + CURRENT_CHAR  
51 51 02 78 0058 219 ASHL #2, R1, R1 ; R1 = 32*RADIX_VALUE  
51 51 53 C0 005C 220 ADDL R3, R1 ; R1 = 40*RADIX_VALUE + CURRENT_CHAR  
005F 221 ; = 50(octal)*RADIX_VALUE + CURRENT_CHAR  
005F 222 :  
005F 223 :  
005F 224 : If any more char to process go back, otherwise return  
005F 225 :  
005F 226 :  
56 D7 005F 227 DECL R6 ; decrement CHARS_REM_IN_WORD by 1  
A2 14 0061 228 BGTR AGAIN1 ; go back to process more char  
05 0063 229 RSB ; return with R1 = RADIX_VALUE  
0064 230 ; R0 = ACTUAL_COUNT  
0064 231 :  
0064 232 ILLEGAL:  
55 D4 0064 233 CLRL R5 ; Illegal character, terminate scan  
05 0066 234 RSB ; Return with R1 = RADIX_VALUE  
0067 235 ; R0 = ACTUAL_COUNT  
0067 236 :  
0067 237 :  
0067 238 .END
```


COM\$SR50WD
Symbol table

J 1
; FORTRAN COMPATIBILITY - ASCII TO RADIX 15-SEP-1984 23:48:32 VAX/VMS Macro V04-00
6-SEP-1984 10:53:21 [FORRTL.SRC]COMR50WD.MAR;1

Page 7
(4)

ACCUM	00000051	R	01
ADD_COUNT	0000004F	R	01
AGAIN1	00000005	R	01
CHECK_DOLLAR	00000039	R	01
CHECK_NUMBER	0000002A	R	01
CHECK_PERIOD	00000043	R	01
COM\$SR50WD_R6	00000000	RG	01
ILLEGAL	00000064	R	01
SPACE	0000004D	R	01

+-----+
! Psect synopsis !
+-----+

PSECT name	Allocation	PSECT No.	Attributes															
ABS	00000000	(0.)	00 (0.)	NOPIC	USR	CON	ABS	LCL	NOSHR	NOEXE	NORD	NOWRT	NOVEC	BYTE				
_F4PCOMPAT\$CODE	00000067	(103.)	01 (1.)	PIC	USR	CON	REL	LCL	SHR	EXE	RD	NOWRT	NOVEC	BYTE				

+-----+
! Performance indicators !
+-----+

Phase	Page faults	CPU Time	Elapsed Time
Initialization	33	00:00:00.09	00:00:00.96
Command processing	116	00:00:00.50	00:00:03.29
Pass 1	67	00:00:00.53	00:00:02.34
Symbol table sort	0	00:00:00.00	00:00:00.00
Pass 2	56	00:00:00.47	00:00:01.82
Symbol table output	2	00:00:00.02	00:00:00.02
Psect synopsis output	3	00:00:00.02	00:00:00.02
Cross-reference output	0	00:00:00.00	00:00:00.00
Assembler run totals	279	00:00:01.63	00:00:08.45

The working set limit was 750 pages.
2859 bytes (6 pages) of virtual memory were used to buffer the intermediate code.
There were 10 pages of symbol table space allocated to hold 9 non-local and 0 local symbols.
238 source lines were read in Pass 1, producing 8 object records in Pass 2.
0 pages of virtual memory were used to define 0 macros.

+-----+
! Macro library statistics !
+-----+

Macro library name	Macros defined
_\$255\$DUA28:[SYSLIB]STARLET.MLB;2	0

0 GETS were required to define 0 macros.

There were no errors, warnings or information messages.

MACRO/ENABLE=SUPPRESSION/DISABLE=(GLOBAL,TRACEBACK)/LIS=LIS\$:COMR50WD/OBJ=OBJ\$:COMR50WD MSRC\$:COMR50WD/UPDATE=(ENH\$:COMR50WD)

0179 AH-BT13A-SE
VAX/VMS V4.0

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY

COMR50WD
LIS

FORDATEDS
LIS

FORDECOMO
LIS

FORB
LIS

COMSETST
LIS

FORASSOC
LIS

FORCLOSEF
LIS

FORDATE
LIS

FORCLOSE
LIS

FORDECOMP
LIS

FORDELETE
LIS

COMRAD50
LIS

COMUSEREX
LIS

FORBITOPS
LIS

FORDEFINE
LIS

FORBACKSP
LIS

FORDISPA
LIS

FORCUTR
LIS